Subject: Re: [PATCH 06/11] sysfs: Implement sysfs tagged directory support.

Posted by Tejun Heo on Tue, 01 Jul 2008 06:47:48 GMT

View Forum Message <> Reply to Message

Hello, Eric.

Eric W. Biederman wrote:

```
>> It's still dynamic from sysfs's POV and I think that will make
>> maintenance more difficult.
>
> Potentially.  I have no problem make it clear that things are more static.
```

Great. :-)

```
>> What you described is pretty much what I'm talking about.  The only
>> difference is whether to use caller-provided pointer as tag or an
>> ida-allocated integer.  The last sentence of the above paragraph is
>> basically sys_tag_enabled() function (maybe misnamed).
>
> So some concrete code examples here.  In the current code in lookup
> what I am doing is:
>
>  tag = sysfs_lookup_tag(parent_sd, parent->d_sb);
>  sd = sysfs_find_dirent(parent_sd, tag, dentry->d_name.name);
>
> With the proposed change of adding tag types sysfs_lookup_tag becomes:
>
> const void *sysfs_lookup_tag(struct sysfs_dirent *dir_sd, struct super_block *sb)
> {
>  const void *tag = NULL;
>
>  if (dir_sd->s_flags & SYSFS_FLAG_TAGGED)
>   tag = sysfs_info(sb)->tag[dir_sd->tag_type];
>
>  return tag;
> }
>
> Which means that in practice I can lookup that tag that I am displaying
> once.
>
> Then in sysfs_find_dirent we do:
>
>  for (sd = parent_sd->s_dir.children; sd; sd = sd->s_sibling) {
>    if ((parent_sd->s_flags & SYSFS_FLAG_TAGGED) &&
>        (sd->s_tag.tag != tag))
>     continue;
>    if (!strcmp(sd->s_name, name))
>     return sd;
```

> }
>
> That should keep the implementation sufficiently inside of sysfs for there
> to be no guessing. In addition as a practical matter we can only allow
> one tag to be visible in a directory at once or else we can not check
> for duplicate names. Which is the problem I see with a bitmap based test
> too unnecessary many degrees of freedom.

Having enumed tag types limits that a sb can have map to only one tag
but it doesn't really prevent multiple possibly visible entries which is
the real unnecessary degrees of freedom. That said, I don't really
think it's an issue.

> The number of tag types will be low as it is the number of subsystems
> that use the feature. Simple enough that I expect statically allocating
> the tag types in an enumeration is a safe and sane way to operate.
> i.e.
>
> enum sysfs_tag_types {
> SYSFS_TAG_NETNS,
> SYSFS_TAG_USERNS,
>   SYSFS_TAG_MAX
> };

I still would prefer something which is more generic. The abstraction
is clearer too. A sb shows untagged and a set of tags. A sd can either
be untagged or tagged (a single tag).

>> The main reason why I'm whining about this so much is because I think
>> tag should be something abstracted inside sysfs proper. It's something
>> which affects very internal operation of sysfs and I really want to keep
>> the implementation details inside sysfs. Spreading implementation over
>> kobject and sysfs didn't turn out too pretty after all.
>
> I agree. Most of the implementation is in sysfs already. We just have
> a few corner cases.
>
> Fundamentally it is the subsystems responsibility that creates the
> kobjects and the sysfs entries. The only case where I can see an
> ida generated number being a help is if we start having lifetime
> issues. Further the extra work to allocate and free tags ida based
> tags seems unnecessary.
>
> I don't doubt that there is a lot we can do better. My current goal
> is for something that is clean enough it won't get us into trouble
> later, and then merging the code. In tree where people can see
> the code and the interactions I expect it will be easier to talk
> about.

>
> Currently the interface with the users is very small. Adding the
> tag_type enumeration should make it smaller and make things more
> obviously static.

Using ida (or idr if a pointer for private data is necessary) is really easy. It'll probably take a few tens of lines of code. That said, I don't think I have enough rationale to nack what you described. So, as long as the tags are made static, I won't object.

Thanks.

--
tejun